GEORGIA INSTITUTE OF TECHNOLOGY
SCHOOL of ELECTRICAL and COMPUTER ENGINEERING

ECE 4550 — Control System Design — Spring 2016
Lab #7: DC Motor Position Control

Contents

1 Background Material

1.1 Introductory Comments

The objective of this lab is to implement a state-space integral control algorithm on a microcontroller, in order to perform position control. We will leverage the reduced-order physics-based model of the DC motor plant; the coefficient matrices of this model will be needed for controller design and implementation, so we will use approximate values inferred from measured input-output data. We will utilize plant model knowledge and the computational capability of the microcontroller to perform the numerical processing required by the state-space integral control algorithm, including the real-time approximate solution of differential equations governing the estimator and regulator subsystems of the controller. We will examine the role of the reference signal used to dictate desired plant behavior, especially the need to employ reference signals smoother than step signals in order to avoid actuator saturation. Finally, we will implement integrator anti-windup compensation to avoid degradation of transient response when actuator saturation cannot be avoided.

1.2 Plant Modeling

Recall from Lab 6 that we typically require two different models of the same DC motor plant. The *simulation model* has three state variables (position, speed and current) whereas the *design* *model* has only two state variables (position and speed). We use the simulation model to represent the plant for simulation purposes, but we use the design model to represent the plant for controller design purposes. We have this option since the dynamics being neglected by the design model—the voltage to current transient response—is a stable and sufficiently fast dynamics.

You should review both the simulation model and the design model from Lab 6 documentation. The simulation model will be explicitly used in Lab 7 simulation code, whereas the design model will be explicitly used for Lab 7 controller design calculations. In order to prepare for the controller design calculations that follow, recall that the design model may be expressed in the form

$$\begin{aligned}\dot{x}(t) &= \begin{bmatrix} 0 & 1 \\ 0 & -\alpha \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ \beta \end{bmatrix} (u(t) - w(t)) \\ y(t) &= \begin{bmatrix} 1 & 0 \end{bmatrix} x(t)\end{aligned}$$

where α and β are load-dependent constant parameters identified from analysis of measured input-output data in Lab 6. The coefficient matrices A, B and C are the point of departure for design.

1.3 Controller Design

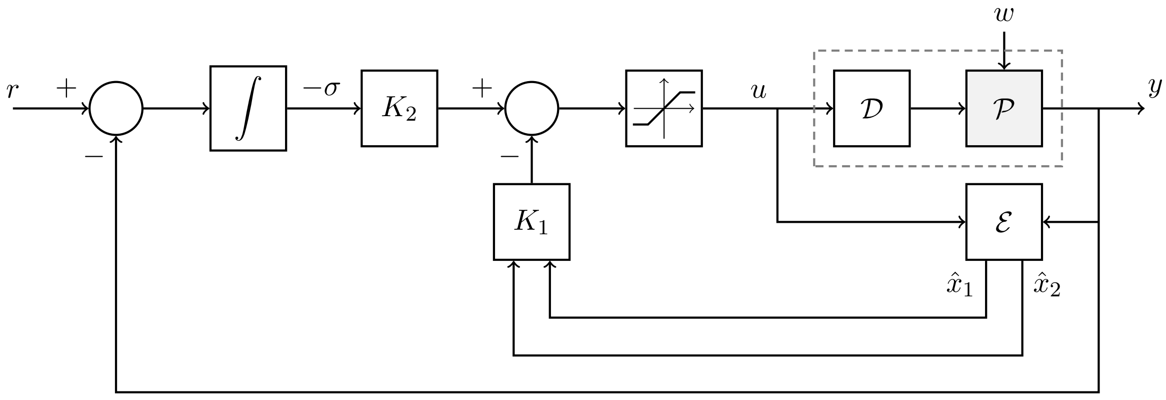
The general theory of state-space integral control has been summarized in some notes posted to tsquare, so *you should have read through those notes prior to attempting this lab*. Critical steps in applying those notes to the control system design problem at hand are highlighted below. Following the design-by-emulation concept, often referred to as "indirect" digital design, we will first design a suitable continuous-time position controller, and then discretize that controller in a final step in order to obtain a corresponding discrete-time position controller.

The controller is implemented in C with GPIO enable signals (Labs 1–2), it operates periodically with period established by a timer interrupt (Lab 3), the internal state-variable signals of the plant are monitored using an analog-to-digital converter (Lab 4), the input signal of the plant is a pulse-width modulated voltage (Lab 5) and the output signal of the plant consists of quadrature encoder pulses (Lab 6). Hence, this lab integrates all that we have learned so far this semester.

1.3.1 Controller Structure

The structure of the controller we will be using in this lab is shown in the block diagram below. The shaded block labeled $\mathcal{P}$ represents our DC motor plant, which consists of an actuator (the drive circuit), a motion system (the rotor) and a sensor (the optical encoder); all other unshaded blocks represent embedded code. The block labeled $\mathcal{D}$ represents the duty cycle generator; it receives a voltage command u and translates that into a duty cycle such that the average voltage applied by the drive circuit will be equal to u. Together, $\mathcal{P}$ and $\mathcal{D}$ constitute a generalized plant. The saturation block accounts for the limits on applied motor voltage inherited as a consequence of the fixed power supply voltage. The external torque disturbance, interpreted as an equivalent external voltage disturbance, is denoted by w. The measured position is denoted by y.

In the inner loop of the controller, the estimator—the block labeled $\mathcal{E}$—generates the scalar signals $\hat{x}_1$ and $\hat{x}_2$, which are estimates of the measured position y and the unmeasured speed $\dot{y}$, respectively. The estimator uses the pre-saturated command voltage as a proxy for the average voltage actually applied to the motor, in order to eliminate the need for any direct measurement of motor voltage. In the outer loop of the controller, the scalar signal σ is generated by integrating the error between the measured position y and its commanded value r. The gain matrices of the estimator (L) and both regulator loops (K_1 and K_2) are selected so as to ensure stabilization of the overall system, and the error integrator guarantees effectively zero steady-state error between the measured position and its commanded value.



1.3.2 Analog Controller Design

From the design model shown above, it is clear that

$$A = \begin{bmatrix} 0 & 1 \\ 0 & -\alpha \end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ \beta \end{bmatrix}, \quad C = \begin{bmatrix} 1 & 0 \end{bmatrix}.$$

The stability of the estimator subsystem is determined by the matrix

$$A - LC = \begin{bmatrix} 0 & 1 \\ 0 & -\alpha \end{bmatrix} - \begin{bmatrix} L_1 \\ L_2 \end{bmatrix} \begin{bmatrix} 1 & 0 \end{bmatrix} = \begin{bmatrix} -L_1 & 1 \\ -L_2 & -\alpha \end{bmatrix}.$$

If we want the actual characteristic polynomial

$$\det\left(sI - (A - LC)\right) = s^2 + (L_1 + \alpha)\, s + (L_1\alpha + L_2)$$

to match the desired characteristic polynomial

$$(s + \lambda_e)^2 = s^2 + 2\lambda_e s + \lambda_e^2$$

having two roots at $s = -\lambda_e$, then the estimator gains must be

$$L_1 = 2\lambda_e - \alpha, \quad L_2 = \lambda_e^2 - 2\alpha\lambda_e + \alpha^2.$$

It was possible to solve the estimator eigenvalue assignment problem in this case because our sensor choice results in an *observable* system. As expected, L_1 and L_2 depend on plant parameters (α, β) and design parameter λ_e. A large λ_e would lead to a fast estimator, but also to large L_1 and L_2 which could result in problems relating to sensor noise amplification; therefore, selection of λ_e involves a trade-off between rate of response and signal/noise ratio.

The regulator subsystem is modeled by the matrices

$$\mathcal{A} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & -\alpha & 0 \\ 1 & 0 & 0 \end{bmatrix}, \quad \mathcal{B} = \begin{bmatrix} 0 \\ \beta \\ 0 \end{bmatrix}, \quad \mathcal{K} = \begin{bmatrix} K_{11} & K_{12} & K_2 \end{bmatrix}$$

and its stability is determined by the matrix

$$\mathcal{A}-\mathcal{B}\mathcal{K}=\begin{bmatrix}0&1&0\\0&-\alpha&0\\1&0&0\end{bmatrix}-\begin{bmatrix}0\\\beta\\0\end{bmatrix}\begin{bmatrix}K_{11}&K_{12}&K_2\end{bmatrix}=\begin{bmatrix}0&1&0\\-\beta K_{11}&-\alpha-\beta K_{12}&-\beta K_2\\1&0&0\end{bmatrix}.$$

If we want the actual characteristic polynomial

$$\det\left(sI-(\mathcal{A}-\mathcal{B}\mathcal{K})\right)=s^3+(\alpha+\beta K_{12})\,s^2+(\beta K_{11})\,s+(\beta K_2)$$

to match the desired characteristic polynomial

$$(s+\lambda_r)^3=s^3+3\lambda_r s^2+3\lambda_r^2 s+\lambda_r^3$$

having three roots at $s=-\lambda_r$, then the regulator gains must be

$$K_{11}=\frac{1}{\beta}3\lambda_r^2,\quad K_{12}=\frac{1}{\beta}(3\lambda_r-\alpha),\quad K_2=\frac{1}{\beta}\lambda_r^3.$$

It was possible to solve the regulator eigenvalue assignment problem in this case because our actuator choice results in a *controllable* system. As expected, K_{11}, K_{12} and K_2 depend on plant parameters (α,β) and design parameter λ_r. A large λ_r would lead to a fast regulator, but also to large K_{11}, K_{12} and K_2 which could result in problems relating to actuator saturation limits; therefore, selection of λ_r involves a trade-off between rate of response and control effort.[1]

Once the gain matrices have been determined, all that would remain for analog control implementation would be to construct an op-amp circuit capable of implementing the feedback law

$$\begin{aligned}u(t)&=-K_1\hat{x}(t)-K_2\sigma(t)\\\dot{\hat{x}}(t)&=A\hat{x}(t)+Bu(t)-L\left(C\hat{x}(t)-y(t)\right)\\\dot{\sigma}(t)&=y(t)-r(t).\end{aligned}$$

Instead, for many reasons, we prefer to approximate this control algorithm on a microcontroller.

1.3.3 Digital Controller Design

By definition, analog controllers are implemented by analog circuits and digital controllers are implemented by digital circuits. Analog controllers provide corrective actions on a continuous-time basis whereas digital controllers provide corrective actions on a discrete-time basis. If we want to implement an existing analog controller on an embedded microcontroller chip, the analog controller must first be discretized to generate a corresponding digital controller. It is not possible to perform *controller discretization* in an exact way, since a discrete-time system response cannot match a continuous-time system response on a continuous-time basis.[2]

For this lab, we will be content to employ the simplest of all discretization schemes, the forward Euler method. This simple method relies on the approximation

$$\dot{x}(kT)\approx\frac{x(kT+T)-x(kT)}{T}$$

[1]A large λ_r could also cause instability, since our controller design neglects current as a state variable.

[2]It is possible to perform *plant discretization* in an exact way, since a continuous-time system response can match a discrete-time system response at sampling instants. This concept is exploited in so-called "direct" digital design.

where x denotes an arbitrary signal, T denotes a fixed sampling period and k represents a discrete-time index. Using this method to approximate derivatives by differences, the previously designed analog controller gets replaced by the corresponding digital controller

$$\begin{aligned} u[k] &= -K_1\hat{x}[k] - K_2\sigma[k] \\ \hat{x}[k+1] &= \hat{x}[k] + T\left(A\hat{x}[k] + Bu[k] - L\left(C\hat{x}[k] - y[k]\right)\right) \\ \sigma[k+1] &= \sigma[k] + T\left(y[k] - r[k]\right). \end{aligned}$$

This digital controller is not equivalent to the analog controller from which it was derived, for the reasons stated above. However, this digital controller will do a good job of emulating the desired analog controller behavior if T is sufficiently small.

To complete the design of our digital controller, we must substitute our plant model matrices (A, B, C) and our selected gain matrices (L, K_1, K_2) into the above equations, we must add the controller saturation effect, and we must exhibit the one-cycle delay introduced by our ISR programming philosophy. The result of these final steps is the computed controller output signal

$$u^*[k] = -K_{11}\hat{x}_1[k-1] - K_{12}\hat{x}_2[k-1] - K_2\sigma[k-1],$$

the saturated controller output signal

$$u[k] = \begin{cases} +U_{\max} & \text{, if } u^*[k] > +U_{\max} \\ -U_{\max} & \text{, if } u^*[k] < -U_{\max} \\ u^*[k] & \text{, otherwise} \end{cases}$$

and the controller state variable update equations

$$\begin{aligned} \hat{x}_1[k+1] &= \hat{x}_1[k] + T\hat{x}_2[k] - TL_1\left(\hat{x}_1[k] - y[k]\right) \\ \hat{x}_2[k+1] &= \hat{x}_2[k] - T\alpha\hat{x}_2[k] + T\beta u[k] - TL_2\left(\hat{x}_1[k] - y[k]\right) \\ \sigma[k+1] &= \sigma[k] + T\left(y[k] - r[k]\right). \end{aligned}$$

These equations have been presented in scalar form to simplify their programming in C.

1.4 Simulation of the Positioning System

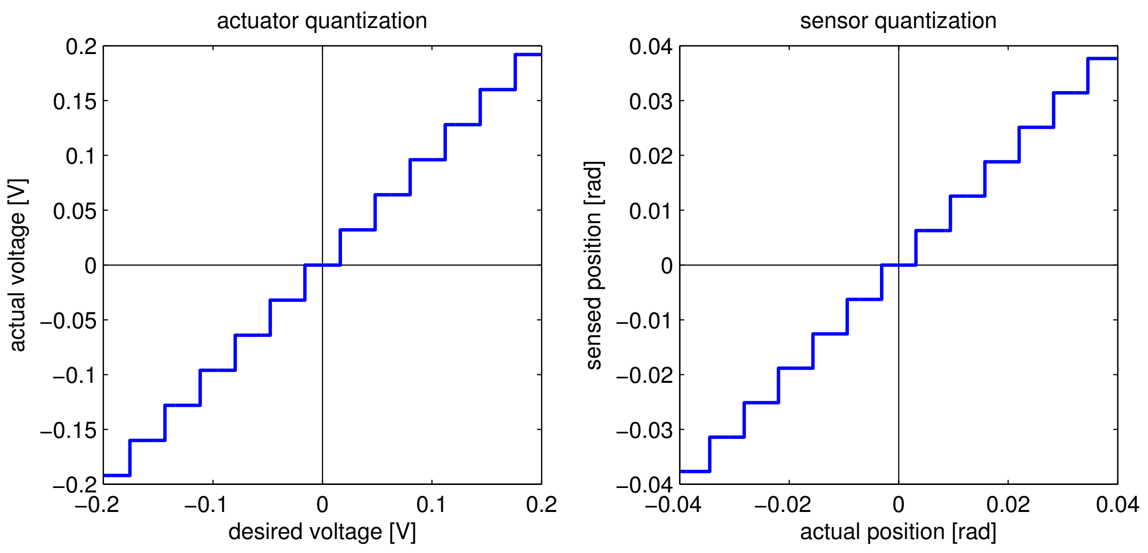
Control engineers follow a systematic process when working on a development project. The process begins with an understanding of the plant physics, including the derivation of its time-domain dynamic model. Selection of an actuator and sensor come next, where the options under consideration should be limited to those that guarantee existence of feedback gains providing internal stability; such guarantees are established from controllability and observability analyses. The next step is to obtain numerical values for all actuator-plant-sensor parameters appearing in the design model, either by deriving these values from physical principles, by finding these values on data sheets, or by evaluating these parameters using experimental measurements. Once the design model parameters are known, the next step is to compute feedback gains. These feedback gains depend on eigenvalue locations typically selected with transient response specifications in mind, but over-ambitious design choices may lead to large feedback gains and consequent problems due to actuator saturation and sensor noise. Hence, *simulation* is an important next step in the design process, especially for microcontroller-based implementations. Simulation provides a convenient way to explore the influence of eigenvalue specification, parameter uncertainty, sampling frequency, and actuator-sensor quantization. We are now at the simulation stage in this project.

1.4.1 Actuator-Sensor Quantization

Before considering the task of simulating the complete control system, let's first learn how to simulate the effect of signal quantization. Our actuator supplies a voltage using 1500 duty cycles over a ±24 V range. Our sensor measures a position, using a 250-slot encoder wheel and 4× decoding logic. The resulting discrepancies between the actuator input signal and actuator output signal, and between the sensor input signal and sensor output signal, are computed in the following Matlab code; both input-output relations are graphically displayed in the figure below.

```
% actuator quantization (1500 duty cycles for +/- 24 V)
Qa = 48/1500;
Xa = 0.2;
xa = Xa*(-1:1/1000:1);
ya = Qa*round(xa/Qa);
subplot(121), plot([-Xa Xa],[0 0],'k',[0 0],[-Xa Xa],'k')
hold on, stairs(xa,ya,'LineWidth',1.5), hold off
axis([-Xa Xa -Xa Xa]), axis square
xlabel('desired voltage [V]')
ylabel('actual voltage [V]')
title('actuator quantization')

% sensor quantization (250 slots per revolution)
Qs = 2*pi/1000;
Xs = 0.04;
xs = Xs*(-1:1/1000:1);
ys = Qs*round(xs/Qs);
subplot(122), plot([-Xs Xs],[0 0],'k',[0 0],[-Xs Xs],'k')
hold on, stairs(xs,ys,'LineWidth',1.5), hold off
axis([-Xs Xs -Xs Xs]), axis square
xlabel('actual position [rad]')
ylabel('sensed position [rad]')
title('sensor quantization')
```



1.4.2 Description of Simulation Code

Now let's consider our main task, which is to simulate the overall system consisting of our continuous-time DC motor plant, with quantized actuator and quantized sensor, under the influence of a discrete-time state-space integral controller commanded by a reference signal. The supplied Matlab code is intended to assist with this task. First you must enter values for the plant parameters. Although nominal values for the unloaded motor may be initially considered, in reality there is generally a mismatch between the characteristics of the true plant and those assumed for sake of controller implementation; this phenomenon is captured by adopting a third-order simulation model and a second-order design model, and by assuming that parameters `J` and `F` represent the true plant, but that `Jhat` and `Fhat` are the parameters actually used for controller implementation; you can vary the `*1` factors to explore the robustness of your control system to parameter mismatch. Next you must enter values for the regulator subsystem eigenvalues, `lambda_r`, and the estimator subsystem eigenvalues, `lambda_e`; typically, we make the estimator eigenvalues faster than the regulator eigenvalues by a factor of four, but you can change the `4*` ratio if desired. Finally, you must enter a value for the sampling period, `T`; 1 ms is a good choice to begin with.

Additional features of this code are as follows: a 24 V power supply is assumed, so the software-generated actuator saturation limits are set to 95% of ± 24 V to avoid current sensing problems; the external disturbance torque is assumed to be zero; the overall system is initialized at the zero equilibrium, with external inputs and state variables all equal to zero; the reference input `r` is preset to command one full revolution; the feedback gain matrices `L`, `K1` and `K2` are computed according to the procedure shown earlier; the forward Euler approximation is used to discretize the controller with period `T` and to simulate the plant with time-step `h`; and a one-cycle delay in application of `u` is assumed, consistent with our standard interrupt-based code flow choices made in previous labs.

1.5 Refinement of the Controller Design

Two modifications of the controller just designed and simulated must be made for practical reasons. Concerns with the controller in its present form are as follows: (i) actuator saturation will lead to integrator windup in any controller with pure integral action, resulting in poor transient performance; (ii) use of abruptly changing step reference signals to command large output transitions will cause actuator saturation to occur. These interrelated issues are resolved by introducing anti-windup integration logic and smoothly-shaped reference commands as described below.

1.5.1 Anti-Windup Compensation

The implementation of state-space integral control as previously described would not perform well if actuator saturation were to occur. The problem, known as integrator windup, has been described in some notes posted to tsquare; generally speaking, integration of output error is not appropriate during time intervals on which the actuator is saturated, since that would lead to an increasing integrator output without resulting in a larger corrective influence.

One approach to anti-windup compensation is to use conditional integration. With this approach, pure integration is performed whenever saturation is not occurring whereas no integration at all is performed whenever saturation is occurring. Assuming that the control signal will be pre-saturated in code so as to avoid triggering saturation within the actuator itself, the synthesis of the control signal involves three steps: (i) compute the desired control signal $u^*(t)$ in the usual way; (ii) pre-saturate $u^*(t)$ to obtain a value for $u(t)$ that is compatible with actuator saturation limits; (iii) compute the integrator output $\sigma(t)$ by assigning either a zero value or the output error value to the integrator input according to the presence or absence of saturation. Mathematically, this three-step procedure is summarized by

$$u^*(t) = -K_1\hat{x}(t) - K_2\sigma(t)$$
$$u(t) = \begin{cases} +U_{\max} & , \text{ if } u^*(t) > +U_{\max} \\ -U_{\max} & , \text{ if } u^*(t) < -U_{\max} \\ u^*(t) & , \text{ otherwise} \end{cases}$$

where

$$\dot{\sigma}(t) = \begin{cases} 0 & , \text{ if } u^*(t) > +U_{\max} \\ 0 & , \text{ if } u^*(t) < -U_{\max} \\ y(t) - r(t) & , \text{ otherwise} \end{cases} .$$

The recommended digital implementation, including the one-cycle delay, is defined by

$$u^*[k] = -K_1\hat{x}[k-1] - K_2\sigma[k-1]$$
$$u[k] = \begin{cases} +U_{\max} & , \text{ if } u^*[k] > +U_{\max} \\ -U_{\max} & , \text{ if } u^*[k] < -U_{\max} \\ u^*[k] & , \text{ otherwise} \end{cases}$$

where

$$\sigma[k+1] = \begin{cases} \sigma[k] & , \text{ if } u^*[k] > +U_{\max} \\ \sigma[k] & , \text{ if } u^*[k] < -U_{\max} \\ \sigma[k] + T\,(y[k] - r[k]) & , \text{ otherwise} \end{cases} .$$

For a more complete discussion of integrator windup and its compensation, including a detailed example with illustrative simulations, please consult the notes posted to tsquare.

1.5.2 Reference Command Shaping

A *feasible* reference command $r(t)$ is one that is consistent with perfect tracking $y(t) \equiv r(t)$ using only unsaturated values of $u(t)$. A simple way of synthesizing feasible reference commands for this lab is suggested by the following diagram, which displays the kinematic relationships between acceleration $\ddot{\theta}(t)$, speed $\dot{\theta}(t)$ and position $\theta(t)$, for the specific case of point-to-point positioning with symmetric piecewise-constant acceleration. Since $y(t) = \theta(t)$ in this lab, the smoothly-shaped plot of $\theta(t)$ in this diagram is understood to represent a feasible reference command $r(t)$ provided that the plant actuator is capable of imposing this motion without saturation.

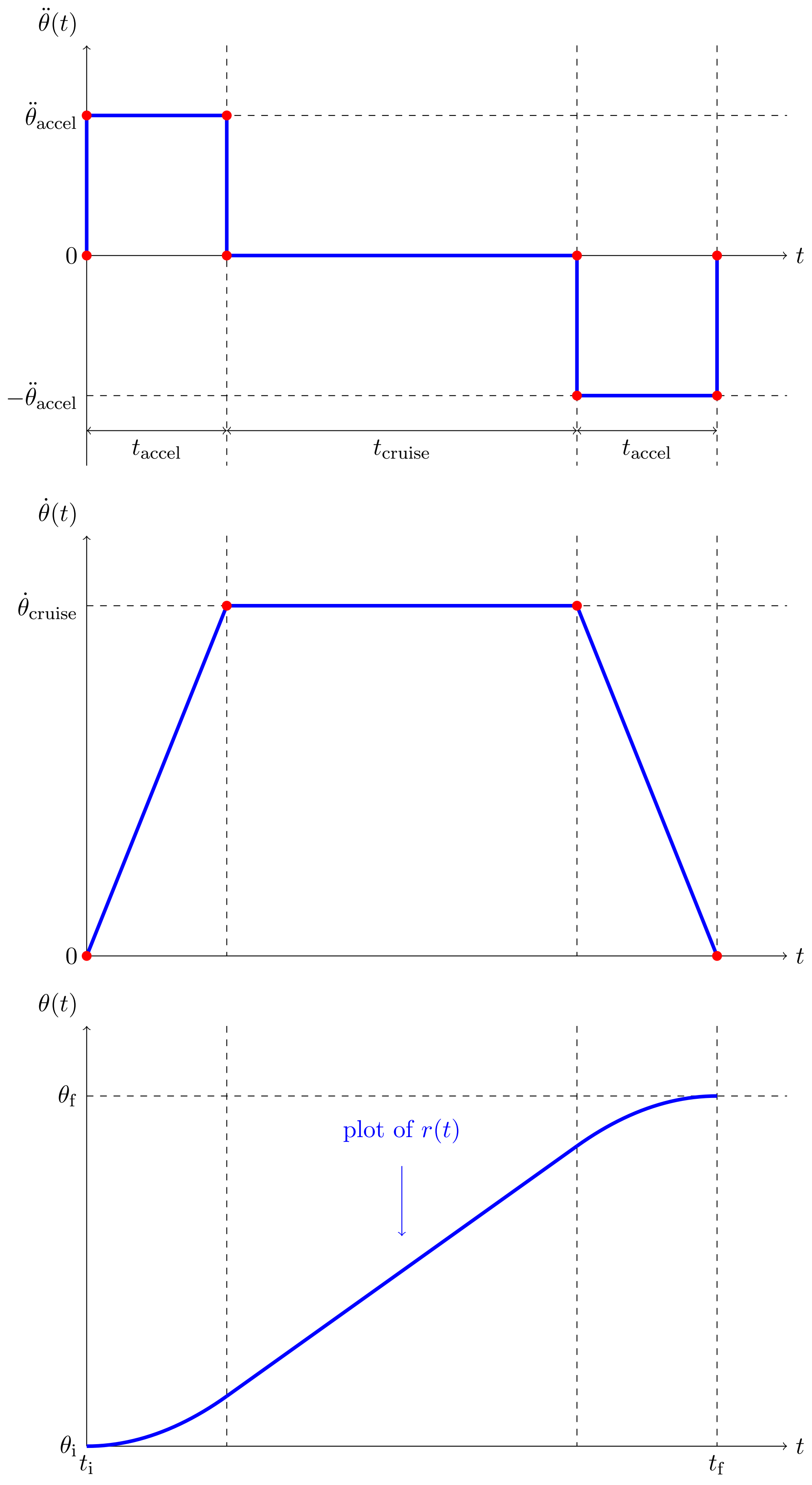
In this motion profile, there is a positive acceleration region, a cruise region, and a negative acceleration region. Both acceleration regions occur on a time interval of length t_{accel}, with constant accelerations equal to $\pm\ddot{\theta}_{\text{accel}}$. The cruise region, which could be omitted if desired, occurs on a time interval of length t_{cruise} and corresponds to a cruise speed equal to $\dot{\theta}_{\text{cruise}}$. Motion begins with position θ_{i} at time t_{i} and ends with position θ_{f} at time t_{f}.

This motion profile is specified in terms of the interrelated coefficients t_{accel}, $\ddot{\theta}_{\text{accel}}$, t_{cruise}, $\dot{\theta}_{\text{cruise}}$, $t_{\text{f}} - t_{\text{i}}$ and $\theta_{\text{f}} - \theta_{\text{i}}$. By visually integrating acceleration to obtain speed, one constraint becomes

$$\dot{\theta}_{\text{cruise}} = \ddot{\theta}_{\text{accel}} t_{\text{accel}}.$$

By visually integrating speed to obtain position, another constraint becomes

$$\theta_{\text{f}} - \theta_{\text{i}} = \dot{\theta}_{\text{cruise}}\,(t_{\text{accel}} + t_{\text{cruise}})\,.$$



$\ddot{\theta}(t)$
$\ddot{\theta}_{\text{accel}}$
0
$-\ddot{\theta}_{\text{accel}}$
t
t_{accel}
t_{cruise}
t_{accel}
$\dot{\theta}(t)$
$\dot{\theta}_{\text{cruise}}$
0
t
$\theta(t)$
θ_{f}
plot of $r(t)$
θ_{i}
t_{i}
t_{f}
t

These constraints imply that the travel time is given by

$$t_\mathrm{f} - t_\mathrm{i} = \frac{\theta_\mathrm{f} - \theta_\mathrm{i}}{\dot\theta_\mathrm{cruise}} + \frac{\dot\theta_\mathrm{cruise}}{\ddot\theta_\mathrm{accel}} > 0,$$

the duration of the acceleration phases is given by

$$t_\mathrm{accel} = \frac{\dot\theta_\mathrm{cruise}}{\ddot\theta_\mathrm{accel}} > 0,$$

and the duration of the cruise phase is given by

$$t_\mathrm{cruise} = \frac{\theta_\mathrm{f} - \theta_\mathrm{i}}{\dot\theta_\mathrm{cruise}} - \frac{\dot\theta_\mathrm{cruise}}{\ddot\theta_\mathrm{accel}} \geq 0.$$

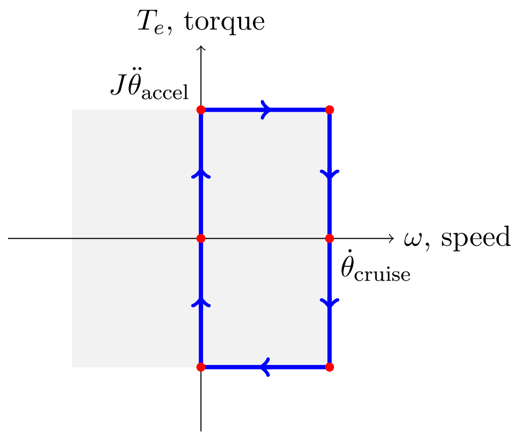
Furthermore, the reference command may be computed according to[3]

$$r(t) = \begin{cases} \theta_\mathrm{i} + \frac{1}{2}\ddot\theta_\mathrm{accel}(t - t_\mathrm{i})^2 & , \ t \in \{\text{positive acceleration time interval}\} \\ \frac{1}{2}\left(\theta_\mathrm{i} + \theta_\mathrm{f}\right) + \dot\theta_\mathrm{cruise}\left(t - \frac{1}{2}\left(t_\mathrm{i} + t_\mathrm{f}\right)\right) & , \ t \in \{\text{cruise time interval}\} \\ \theta_\mathrm{f} - \frac{1}{2}\ddot\theta_\mathrm{accel}(t_\mathrm{f} - t)^2 & , \ t \in \{\text{negative acceleration time interval}\} \end{cases}$$

where t_i, t_f, θ_i, θ_f, $\dot\theta_\mathrm{cruise}$ and $\ddot\theta_\mathrm{accel}$ appear explicitly and t_accel and t_cruise appear implicitly.

A motion profile of this type is feasible if the plant actuator has the capability of imposing the profile-specified accelerations at the corresponding profile-specified speeds. Our plant has limited acceleration (due to limited current) and limited speed (due to limited voltage); typically these limitations are characterized by a torque-speed capability region. One task of the control engineer is therefore to assign the motion profile parameters t_accel and t_cruise in terms of limits on $\ddot\theta_\mathrm{accel}$ and $\dot\theta_\mathrm{cruise}$, such that a motion from position θ_i at time t_i to position θ_f at time t_f becomes feasible.

How do we determine which smoothly-shaped reference commands are feasible and which are not feasible? To answer this question, it is necessary to relate the torque-speed capability region of the actuator to the reference command trajectory traced out in the torque-speed plane. Such a trajectory is displayed below, wherein the red dots correspond to the critical points that must be located within the torque-speed capability region of the actuator; J is the total inertia of the rotor and load, and friction is neglected. The blue trajectory tracing out a rectangle in the right-half plane is associated with increasing positions, whereas a similar region in the left-half plane would be associated with decreasing positions. This analysis establishes that the shaded rectangular region shown below, having vertexes of magnitude $(\dot\theta_\mathrm{cruise}, J\ddot\theta_\mathrm{accel})$, must fit inside the torque-speed capability region of the actuator if this family of reference command trajectories is to be feasible.



$T_e = J\ddot\theta$ neglects friction

red dots correspond with critical points of trajectory (see the plots on page 9)

[3] In this formula, $\ddot\theta_\mathrm{accel}$ and $\dot\theta_\mathrm{cruise}$ have been implicitly assumed to be positive (for forward motion); the reverse motion case corresponds to negative values for these two parameters.

How do we determine the torque-speed capability region of the actuator? There are two limits to consider. To maintain safe operation we require that motor current be no larger than some constant value $I_{\max}$, and from our earlier study of the power converter circuit we know that (average) motor voltage can be no larger than the supply voltage V_{dc}. Using motor model notation from Lab 6, motor current i and (average) motor voltage v must satisfy

$$|i| \leq I_{\max}, \quad |v| \leq V_{\text{dc}}.$$

The resistive voltage must be overcome if the current limit is to be reached at zero speed, so

$$V_{\text{dc}} > RI_{\max}.$$

Induced torque is proportional to current, so one bound on torque capability is given by

$$T_e = Ki \quad \Rightarrow \quad T_e \leq KI_{\max}, \text{ in 1st quadrant.}$$

Another bound on torque capability may be derived by considering the relationship between motor current and motor voltage, simplified by assuming either that inductance L is negligibly small or that the rate-of-change of current di/dt is negligibly small; the relationship $v = Ri + K\omega$ exhibits the influence of induced voltage, which is proportional to speed. Eliminating i from this relationship by substituting $T_e = Ki$, another bound on torque capability is given by

$$T_e = \frac{K}{R}(v - K\omega) \quad \Rightarrow \quad T_e \leq \frac{K}{R}(V_{\text{dc}} - K\omega), \text{ in 1st quadrant.}$$

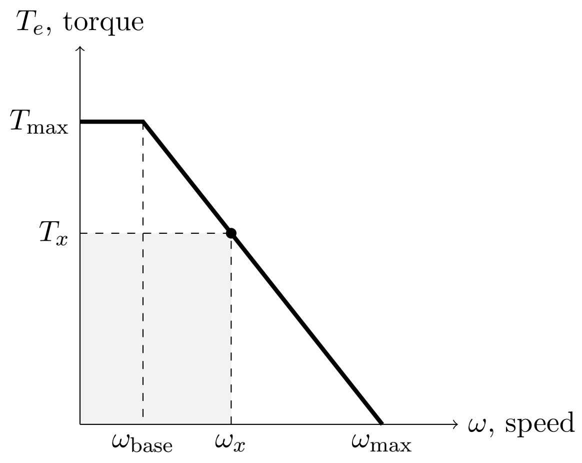
Taking the intersection of these two sets, the boundary of the torque-speed capability region is as shown below; only the 1st quadrant is displayed, since this provides sufficient information for specifying feasible reference command trajectories. Some of the critical torque and speed values labeled on the axes of this figure are easily shown to be

$$T_{\max} = KI_{\max}, \quad \omega_{\text{base}} = \frac{V_{\text{dc}} - RI_{\max}}{K}, \quad \omega_{\max} = \frac{V_{\text{dc}}}{K}.$$

Of greatest interest is the shaded rectangular region with vertex (ω_x, T_x) where

$$T_x = \frac{K}{R}(V_{\text{dc}} - K\omega_x), \quad \omega_{\text{base}} \leq \omega_x \leq \omega_{\max}.$$

A feasible reference command trajectory requires that $\dot{\theta}_{\text{cruise}} \leq \omega_x$ and $J\ddot{\theta}_{\text{accel}} \leq T_x$.



Now we determine the smallest feasible travel time $t_{\text{travel}} = t_{\text{f}} - t_{\text{i}}$ corresponding to a desired travel distance $\theta_{\text{travel}} = \theta_{\text{f}} - \theta_{\text{i}}$. Assigning reference command trajectory parameters so that

$$\dot{\theta}_{\text{cruise}} = \omega_x, \quad J\ddot{\theta}_{\text{accel}} = T_x,$$

the travel time may be expressed solely as a function of the free parameter ω_x:

$$t_{\text{travel}} = \frac{JR\omega_x}{K(V_{\text{dc}} - K\omega_x)} + \frac{\theta_{\text{travel}}}{\omega_x}.$$

To minimize travel time, we differentiate to impose

$$\frac{\partial t_{\text{travel}}}{\partial \omega_x} = 0$$

which then leads to the quadratic constraint

$$\left(K^3\theta_{\text{travel}} - V_{\text{dc}}JR\right)\omega_x^2 - \left(2V_{\text{dc}}K^2\theta_{\text{travel}}\right)\omega_x + \left(V_{\text{dc}}^2K\theta_{\text{travel}}\right) = 0.$$

Applying the quadratic formula, the desired torque-speed vertex coordinates are

$$\omega_x = V_{\text{dc}}\left(\frac{K^2\theta_{\text{travel}} - \sqrt{V_{\text{dc}}JRK\theta_{\text{travel}}}}{K^3\theta_{\text{travel}} - V_{\text{dc}}JR}\right)$$
$$T_x = V_{\text{dc}}\frac{K}{R}\left(\frac{K\sqrt{V_{\text{dc}}JRK\theta_{\text{travel}}} - V_{\text{dc}}JR}{K^3\theta_{\text{travel}} - V_{\text{dc}}JR}\right).$$

The voltage limit V_{dc} appears here rather than the current limit $I_{\max}$, since we have chosen to focus on the higher-speed triangular portion of the 1st-quadrant torque-speed capability region. This analysis demonstrates how reference command shaping is a nontrivial parameter optimization problem that requires explicit knowledge of plant physics and involves various cases.

2 Lab Assignment

2.1 Pre-Lab Preparation

Each individual student must work through the pre-lab activity and prepare a pre-lab deliverable to be submitted *by the beginning of the lab session*. The pre-lab deliverable consists of a brief typed statement, no longer than one page, in response to the following pre-lab activity specification:

1. Read through this entire document, and describe the overall purpose of this week's project.
2. Simulate the control system using the supplied code, in order to provide informed responses to the following questions; use words to convey your observations (don't submit plots).
   (a) How were the formulas for `alpha` and `beta` derived (see code lines 16 and 17)?
   (b) What happens when `Jhat` and `Fhat` are not equal to `J` and `F`?
   (c) What happens when `lambda_r` and `lambda_e` are varied?
   (d) What happens when `T` is varied?
   (e) What happens when `r` is varied?
   (f) How big is the steady-state positioning error?

Please note that it is not essential to write application code prior to the lab session; the point of the pre-lab preparation is for you to arrive at the lab session with a clear understanding of how the plant should respond to the controller we will implement during the lab session.

2.2 Specification of the Assigned Tasks

2.2.1 Position Control with Anti-Windup Compensation

Develop code that implements the integral controller of §1.3.3, including the refinement of §1.5.1; incorporate the possibility to disable or enable the anti-windup compensation, in order to easily make comparisons. An *abruptly changing squarewave* position reference signal should be generated in code to request periodic transitions between two programmable positions; for context, imagine that your motor is connected to a belt-pulley apparatus that will locate a robotic pick-and-place tool on a linear motion axis in order to automate the assembly of a printed circuit board. The PWM, QEP and ADC modules should be configured as in Lab 6.

1. Set the programmable positions to 0 rad and 2π rad, and request transitions between these positions once per half second; the complete forward-reverse cycle will take 1 second. Assign the regulator parameter λ_r so as to provide the fastest transient response possible without saturating the actuator. Since actuator saturation should not occur in this case, anti-windup compensation need not be enabled.

2. Set the programmable positions to 0 rad and 10π rad, and request transitions between these positions once per second; the complete forward-reverse cycle will take 2 seconds. Use the value of λ_r obtained from the previous case; this choice will induce significant actuator saturation due to the larger motions involved. Observe the system response with and without anti-windup compensation enabled.

For Case 1 and Case 2, perform data logging of u (in V), y (in rad) and i (in A). For Case 2 only, export these data, generate response plots in Matlab, and summarize how the presence or absence of anti-windup compensation has influenced the response of the system.

Instructor Verification (separate page)

2.2.2 Position Control with Shaped Reference Command

Develop code that implements the integral controller of §1.3.3, including the refinement of §1.5.2, enabling large motion without actuator saturation. A *feasible smoothly-shaped* position reference signal should be generated in code to request periodic transitions between two programmable positions; for context, imagine that your motor is connected to a belt-pulley apparatus that will locate a robotic pick-and-place tool on a linear motion axis in order to automate the assembly of a printed circuit board. The PWM, QEP and ADC modules should be configured as in Lab 6.

Set the programmable positions to 0 rad and 10π rad, and set the travel time between these positions to be as small as possible without exceeding the torque-speed capability limits of the plant actuator, assuming $V_{\text{dc}} = 24$ V and $I_{\max} = 2.5$ A. Set the reference command parameters using the formulas supplied on page 12; since this requires load information, provide equations showing how you can compute the load-dependent values J and F from the identified values of α and β and the data sheet values of K and R. Include a short dwell time (see below) at both 0 rad and 10π rad to reflect the requirements of the hypothetical pick-and-place application.

1. At $t = 0.0$ s, begin the smoothly-shaped r from $r = 0$ rad to $r = 10\pi$ rad.

2. Due to optimized parameter choices, this motion will complete prior to $t = 0.5$ s.

3. At $t = 0.5$ s, begin the smoothly-shaped r from $r = 10\pi$ rad to $r = 0$ rad.

4. Due to optimized parameter choices, this motion will complete prior to $t = 1.0$ s.
5. Repeat the 1.0 s back-and-forth cycle indefinitely.

This amount of dwell time will provide sufficient time for y to converge to r at the end of each motion, without holding at $r = 0$ rad or $r = 10\pi$ rad for so long that data logging would take up too much memory. Log voltage u (in V), current i (in A), reference position r (in rad) and actual position y (in rad) for one complete motion cycle from 0 rad to 10π rad and back to 0 rad (i.e. 1 s duration). Generate response plots in Matlab using exported data, and summarize your findings.

Instructor Verification (separate page)

GEORGIA INSTITUTE OF TECHNOLOGY
SCHOOL of ELECTRICAL and COMPUTER ENGINEERING

ECE 4550 — Control System Design — Spring 2016

Lab #7: DC Motor Position Control

Instructor Verification Page

Lab Section	Begin Date	End Date
L01, L02	March 8	March 15
L03, L04	March 10	March 17

To be eligible for full credit, do the following:

1. Submissions required by each student (one per student)
   (a) Upload your pre-lab deliverable to tsquare before lab session begins on begin date.
   (b) Upload your `main.c` file for §2.2.2 to tsquare before lab session ends on end date.
2. Submissions required by each group (one per group)
   (a) Submit a hard-copy of this verification page before lab session ends on end date.
   (b) Attach to this page the hard-copy plots and summaries requested in §2.2.1 and §2.2.2.

Name #1: ________________________________

Name #2: ________________________________

Checkpoint: Verify completion of the task assigned in §2.2.1.

Verified: ______________________ **Date/Time:** ______________________

Checkpoint: Verify completion of the task assigned in §2.2.2.

Verified: ______________________ **Date/Time:** ______________________